Hums & whistles melody recognition

Hummingbird

Abstract

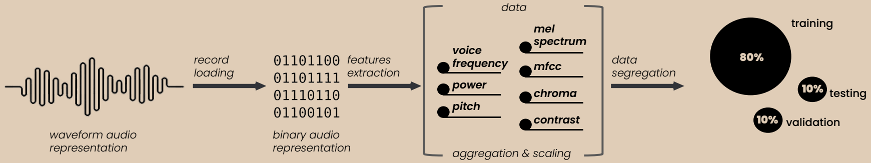
How many times has it happened that we hear a song for the first time, but forget to find it right away?

In order to find that song, the easiest way is to know the lyrics or at least part of the lyrics. However, that is rarely the case after the first listen. Melody, on the other hand, is much easier to remember. Therefore, it would be more convenient to find the song by humming or whistling a memorized melody..

The goal of this project is to make ML model that will try to identify song from dataset that contains eight different melodies mad with hum or whistle. Something similar to what Shazam offers.

Algorithms

Data is first preprocessed. During that process, we extract important features like *pitch*, *power*, *voiced frequency*, *mel spectrogram*, *mel frequency cepstral coefficients*, *chroma* and *contrast*. Such processed data is then scaled and split into three sets: training (80%), validation (10%) and testing (10%) data.



Now, we decided to create two different models and compare their performances:

1. Support Vector Classifier (SVC)
2. Convolutional Neural Network (CNN)

Authors:
Nemanja Dutina & Milica Sladaković

Course:
Computational Intelligence

Mentor:
Branislav Anđelković

Professor:
dr Aleksandar Kovačević

https://github.com/coma007/hummingbird/`

Results

SVC model was trained with a regularization parameter *C = 1*. The model was evaluated using testing and validation datasets, and it achieved an impressive accuracy of *99%* on both datasets. This high accuracy suggests that the SVC model is performing exceptionally well in classifying the unseen data, with only a very small percentage of misclassifications.

It's important to note that such high accuracy may indicate a good fit of the model, but it is also essential to consider potential *overfitting*. However, testing model again on larger dataset and maintaining its accuracy, allowed us to conclude that model has trained very well and can generalize its knolewdge to classify unseen data.

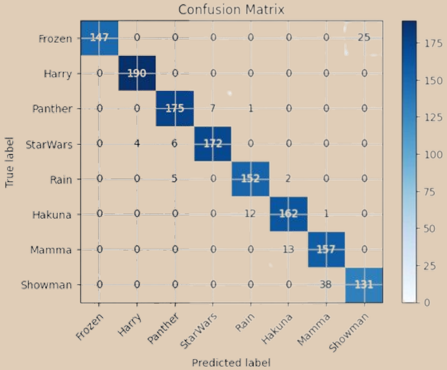


Image on the left shows confusion matrix of cnn model on larger dataset.

CNN model was trained within *60 epochs*, *batch size* equal to *3* and *Adam optimizer* (with *learning rate of 0.001*, **β** *1 = 0.9* and **β** *2 = 0.999*). Model was defined as sequential model with *6 dense layers*. Every layer had half as many units as the previous one with the first one having *1024* and the last one *8* (equal to number of classes). Every layer used ***ReLu* actiovation function**, except last layer, where **softmax** was used. We modified different parameters like number of epochs, batch size, learning rate, betas and layers. By increasing the epochs or batch size, we ended up with model that was progressing at much slower rate. If we changed learning rate or betas, model was underperforming with very low accuracy. This model achieved accuracy of *91.7%* in both testing and validation phase.

RFC model was trained with *10k-20k estimators*. During training, model achieved training accuracy of 100%, indicating that it performed extremely well on the training data. We concluded that we cannot influence possible model overfiting with number of estimators, but that models trained on larger datasets (about 4000 samples) perform much better, with accuracy of *74.85%* in validation phase. The high testing accuracy of 100% indicates that the model memorized the training data, including its noise and outliers. This level of overfitting is causing poor performance on the validation data, as the model fails to capture the true underlying patterns of the data.